Relational Collaborative Topic Regression for Recommender Systems

Hao Wang and Wu-Jun Li, *Member, IEEE*

Abstract—Due to its successful application in recommender systems, collaborative filtering (CF) has become a hot research topic in data mining and information retrieval. In traditional CF methods, only the feedback matrix, which contains either explicit feedback (also called ratings) or implicit feedback on the items given by users, is used for training and prediction. Typically, the feedback matrix is sparse, which means that most users interact with few items. Due to this sparsity problem, traditional CF with only feedback information will suffer from unsatisfactory performance. Recently, many researchers have proposed to utilize auxiliary information, such as item content (attributes), to alleviate the data sparsity problem in CF. Collaborative topic regression (CTR) is one of these methods which has achieved promising performance by successfully integrating both feedback information and item content information. In many real applications, besides the feedback and item content information, there may exist relations (also known as networks) among the items which can be helpful for recommendation. In this paper, we develop a novel hierarchical Bayesian model called Relational Collaborative Topic Regression (RCTR), which extends CTR by seamlessly integrating the user-item feedback information, item content information, and network structure among items into the same model. Experiments on real-world datasets show that our model can achieve better prediction accuracy than the state-of-the-art methods with lower empirical training time. Moreover, RCTR can learn good interpretable latent structures which are useful for recommendation.

Index Terms—Collaborative filtering, topic models, recommender system, social network, relational learning

1 Introduction

Recommender systems (RS) play an important role to enable us to make effective use of information. For example, Amazon [28] adopts RS for product recommendations, and Netflix [8] uses RS for movie recommendations. Existing RS methods can be roughly categorized into three classes [1], [4], [10], [41]: content-based methods, collaborative filtering (CF) based methods, and hybrid methods. Content-based methods [5], [26] adopt the profile of the users or products for recommendation. CF based methods [7], [11], [13], [18], [27], [30], [33], [35], [37], [39], [42] use past activities or preferences, such as the ratings on items given by users, for prediction, without using any user or product profiles. Hybrid methods [2], [3], [6], [12], [15], [16], [17], [20], [32], [34], [40], [43], [47], [48] combine both content-based methods and CF based methods by ensemble techniques. Due to privacy issues, it is harder in general to collect user profiles than past activities. Hence, CF based methods have become more popular than content-based methods in recent years.

In most traditional CF methods, only the feedback matrix, which contains either explicit feedback (also called ratings) or implicit feedback [21] on the items given by users, is used for training and prediction. Typically, the feedback matrix is sparse, which means that most items are given feedback by few users or most users only give feedback to few items. Due to this sparsity problem, traditional CF with only feedback information will suffer from unsatisfactory performance. More specifically, it is difficult for CF methods to achieve good performance in both item-oriented setting and user-oriented setting when the feedback matrix is sparse. In an item-oriented setting where we need to recommend users to items, it is generally difficult to know which users could like an item if it has only been given feedback by one or two users. This adds to the difficulty companies face when promoting new products (items). Moreover, users' ignorance of new items will result in less feedback on the new items, which will further harm the accuracy of their recommendations. For the user-oriented setting where we recommend items to users, it is also difficult to predict what a user likes if the user has only given feedback to one or two items. However, in the real world, it is common to find that most users provide only a little feedback. Actually, providing good recommendations for new users with little feedback is more important than for frequent users since new users will only come back to the site (service) depending on how good the recommendation is. However, for frequent users, it is most likely that they are already satisfied with the site (service). If we manage to boost the recommendation accuracy for new or infrequent users, more of them will become frequent users, and then better recommendations can be expected with more training data. Therefore, improving the recommendation accuracy at an extremely sparse setting is key to getting the recommender systems working in a positive cycle.

To overcome the sparsity problem of CF-based models, many researchers have proposed to integrate auxiliary

- *H. Wang is with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong. E-mail: hwangaz@cse.ust.hk.*
- *W.-J. Li is with the National Key Laboratory of Novel Software Technology, Department of Computer Science and Technology, Nanjing University, Nanjing 210023, China. E-mail: liwujun@nju.edu.cn.*

Manuscript received 24 July 2013; revised 29 Aug. 2014; accepted 6 Oct. 2014. Date of publication 29 Oct. 2014; date of current version 27 Mar. 2015. Recommended for acceptance by Y. Koren.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2014.2365789

information into the model training and prediction procedures. Some methods [45], [51] utilize the item content (attributes) to facilitate the CF training. One representative of these methods is collaborative topic regression (CTR) [45] which jointly models the user-item feedback matrix and the item content information (texts of articles). CTR seamlessly incorporates topic modeling [9] with CF to improve the performance and interpretability. For new items, CTR is able to perform out-of-matrix prediction (cold-start prediction) [38], [51] using only the content information. Some other methods [22], [29], [31] try to use social networks among *users* to improve the performance. Among these methods, CTR-SMF [31] extends CTR by integrating the social network among users into CTR with social matrix factorization (SMF) [29] techniques, which has achieved better performance than CTR.

In many real applications, besides the feedback and item content information, there may exist relations (or networks) among the *items* [44], [46] which can also be helpful for recommendation. For example, if we want to recommend papers (references) to users in CiteULike,[1] the citation relations between papers are informative for recommending papers with similar topics. Other examples of item networks can be found in hyperlinks among webpages, movies directed by the same directors, and so on.

In this paper, we develop a novel hierarchical Bayesian model, called Relational Collaborative Topic Regression (RCTR), to incorporate *item relations* for recommendation. The main contributions of RCTR are outlined as follows:

- By extending CTR, RCTR seamlessly integrates the user-item feedback information, item content information and relational (network) structure among items into a principled hierarchical Bayesian model.
- Even if a new item has been given feedback only by one or two users, RCTR can make effective use of the information from the item network to alleviate the data sparsity problem in CF, which will consequently improve the recommendation accuracy.
- In cases where a new user has given feedback to only one or two items, RCTR can also make effective use of the information from the item network to improve the recommendation accuracy.
- In RCTR, a family of link (relation) probability functions is proposed to model the relations between items. This extension from discrete link probability functions like those in [14] to a family of link probability functions increases the modeling capacity of RCTR with better performance.
- Compared with CTR, a smaller number of learning iterations are needed for RCTR to achieve satisfactory accuracy. As a consequence, the total empirical measured runtime of training RCTR is lower than that of training CTR even if the time complexity of each RCTR iteration is slightly higher than that of CTR.
- RCTR can learn good interpretable latent structures which are useful for recommendation.
- Experiments on real-world datasets show that RCTR can achieve higher prediction accuracy than the state-of-the-art methods.

Note that CTR-SMF [31] tries to integrate the *user relations* into the CTR model, which is different from the application scenarios of our RCTR model.

The rest of this paper is organized as follows. In Section 2, we briefly introduce the background of CF and CTR. Section 3 presents the details of our proposed RCTR model. Experimental results are described in Section 4 and finally Section 5 concludes the whole paper.

2 Background

In this section, we give a brief introduction about the background of RCTR, including CF based recommendation, matrix factorization (MF) (also called latent factor model) based CF methods [25], [35], [36], [49], [50], and CTR [45].

2.1 CF Based Recommendation

The task of CF is to recommend items to users based on their past feedback of the users. For example, we can deploy a recommender system to recommend papers (references) to researchers in CiteULike. Here users are researchers and items are papers. As in [45], we assume there are I users and J items in this paper. The feedback on item j given by user i is denoted by r_{ij}. Although our model is general enough to be used for other settings with explicit feedback such as the case with integer ratings ranging from 1 to 5, we assume $r_{ij} \in \{0, 1\}$ in this paper which is the same setting as that in CTR [45]. Note that this is a setting with implicit feedback as introduced in [21]. This means that our model tries to predict whether a user likes a item or not. In training data, $r_{ij} = 1$ means that user i likes item j. $r_{ij} = 0$ means that the element is unobserved (missing), i.e., we do not know whether user i likes item j or not. As stated in Section 1, CF methods use only the feedback matrix $\{r_{ij} | i = 1, 2, \ldots, I; j = 1, 2, \ldots, J\}$ for training and prediction.

There are two different cases of prediction [45]: in-matrix prediction and out-of-matrix prediction. For the item-oriented setting, in-matrix prediction tries to make recommendation for items with at least one feedback from the users in the training data. On the contrary, out-of-matrix prediction tries to make recommendation for items without any feedback in the training data. The in-matrix prediction and out-of-matrix prediction for user-oriented settings are similar except that we make recommendation for users rather than items. Out-of-matrix prediction is actually the so-called cold-start recommendation in some of the literature [38], [51].

2.2 Matrix Factorization for CF

The existing CF methods can be divided into two main categories [24]: memory-based methods [18], [28], [37] and model-based methods [19], [25], [35]. Memory-based methods adopt the (weighted) average of the feedback of similar (neighborhood) users or items for prediction, while model-based methods try to learn a statistical model from the training data. Many works have verified that model-based methods can outperform memory-based methods in general. Hence, model-based methods have become more popular in recent years.

1. http://www.citeulike.org/

Matrix factorization [25], [35] and its extensions, such as the probabilistic matrix factorization (PMF) [35], are the most representative model-based methods which in practice have achieved promising performance. The basic idea of MF is to use latent vectors in a low-dimensional space to represent the users and items. More specifically, user i is represented by a latent vector $u_i \in R^K$ of dimensionality K, and item j is represented by a latent vector $v_j \in R^K$. The prediction of the feedback on item j given by user i can be computed as follows:

$$\hat{r}_{ij} = u_i^T v_j.$$

If we use two latent matrices $U = (u_i)_{i=1}^I$ and $V = (v_j)_{j=1}^J$ to denote the latent vectors for all the users and items respectively, it means MF has learnt to find the optimal U and V by optimizing the following objective function:

$$\min_{U,V} \sum_{i=1}^{I} \sum_{j=1}^{J} \left(r_{ij} - u_i^T v_j\right)^2 + \lambda_u \sum_{i=1}^{I} \|u_i\|^2 + \lambda_v \sum_{j=1}^{J} \|v_j\|^2, \quad (1)$$

where $\|\cdot\|$ denotes the Frobenius norm of a vector, λ_u and λ_v are regularization terms for controlling model complexity. The objective function in (1) corresponds to the maximum a posteriori (MAP) estimate of the PMF model in [35].

In [45], a generalization of PMF model is proposed

$$\begin{aligned} u_i &\sim \mathcal{N}(0, \lambda_u^{-1} I_K), \\ v_j &\sim \mathcal{N}(0, \lambda_v^{-1} I_K), \\ r_{ij} &\sim \mathcal{N}(u_i^T v_j, c_{ij}^{-1}), \end{aligned} \quad (2)$$

where $\mathcal{N}(\cdot)$ denotes the normal distribution, I_K is an identity matrix with K rows and columns, and c_{ij} is defined as follows:

$$c_{ij} = \begin{cases} a, & \text{if } r_{ij} = 1, \\ b, & \text{if } r_{ij} = 0, \end{cases}$$

with a and b being tuning parameters and $a > b > 0$. Please note that all the (i, j) pairs with feedback 0 in the training set are used for training in this paper. We can also sample part of them for training in cases where the number of zeros in the feedback matrix is too large.

MF methods have achieved promising performance in practice. However, they also suffer from the sparsity problem. Furthermore, as stated in [45], it is not easy for MF methods to perform out-of-matrix prediction.

2.3 Collaborative Topic Regression

Collaborative topic regression [45] is proposed to recommend documents (papers) to users by seamlessly integrating both feedback matrix and item (document) content information into the same model, which can address the problems faced by MF based CF. By combining MF and latent Dirichlet allocation (LDA) [9], CTR achieves better prediction performance than MF based CF with better interpretable results. Moreover, with the item content information, CTR can predict feedback for out-of-matrix items.

The graphical model of CTR is shown in Fig. 1.

CTR introduces an item latent offset ϵ_j between the topic proportions θ_j in LDA and the item latent vectors v_j in CF. The offset can be explained by the gap between what the article is about (represented by θ_j) and what the users think of it (represented by v_j), as discussed in [45]. If we use $\boldsymbol{\beta} = \beta_{1:K}$ to denote K topics, the generative process of CTR can be listed as follows:

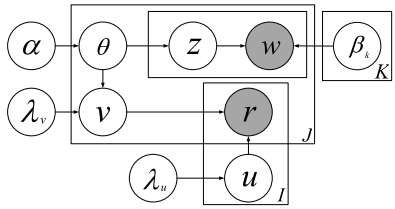


Fig. 1. The graphical model of collaborative topic regression.

1) Draw a user latent vector for each user i: $u_i \sim \mathcal{N}(0, \lambda_u^{-1} I_K)$.
2) For each item j,
   a) Draw topic proportions $\theta_j \sim \text{Dirichlet}(\alpha)$.
   b) Draw item latent offset $\epsilon_j \sim \mathcal{N}(0, \lambda_v^{-1} I_K)$, then set the item latent vector to be: $v_j = \epsilon_j + \theta_j$.
   c) For each word in the document (item) $\mathbf{w}_j$, which is denoted as w_{jn},
      i) Draw topic assignment $z_{jn} \sim \text{Mult}(\theta_j)$.
      ii) Draw word $w_{jn} \sim \text{Mult}(\beta_{z_{jn}})$.
3) Draw the feedback r_{ij} for each user-item pair (i, j),

$$r_{ij} \sim \mathcal{N}(u_i^T v_j, c_{ij}^{-1}).$$

As mentioned in [45], the key to CTR lies in the item latent offset ϵ_j, which makes the item latent vector v_j close enough to the topic proportions θ_j and diverge from it if necessary. Parameter λ_v controls how close v_j is to θ_j.

Experiments on scientific article recommendation from CiteULike show that CTR can outperform MF based CF methods.

3 Relational Collaborative Topic Regression

In this section, we describe the details of our proposed model, called Relational Collaborative Topic Regression. Besides the feedback and item content information modeled by CTR, RCTR can also model the relations among the items which are informative for recommendations.

3.1 Model Formulation

To better illustrate the graphical model of RCTR, we adopt a way different from that in Fig. 1 which is adopted by the authors of CTR [45]. The graphic model of RCTR is shown in Fig. 2, in which the component in the dashed rectangle is what differentiates RCTR from CTR.

The generative process of RCTR is as follows:

1) Draw a user latent vector for each user i: $u_i \sim \mathcal{N}(0, \lambda_u^{-1} I_K)$.
2) For each item j,
   a) Draw topic proportions $\theta_j \sim \text{Dirichlet}(\alpha)$.
   b) Draw item latent offset $\epsilon_j \sim \mathcal{N}(0, \lambda_v^{-1} I_K)$, then set the item latent vector to be: $v_j = \epsilon_j + \theta_j$.
   c) Draw item relational offset $\tau_j \sim \mathcal{N}(0, \lambda_r^{-1} I_K)$, then set the item relational vector to be: $s_j = \tau_j + v_j$.

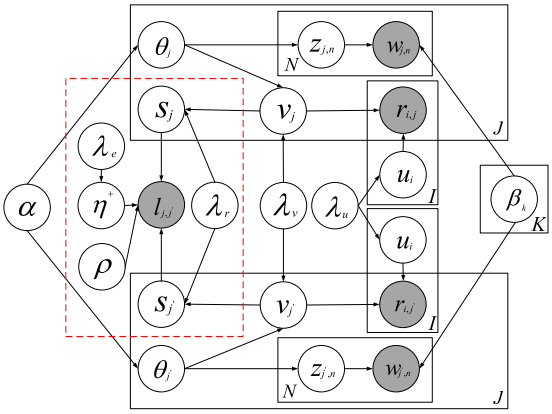


Fig. 2. The graphical model of RCTR. The component in the dashed rectangle is what differentiates RCTR from CTR.

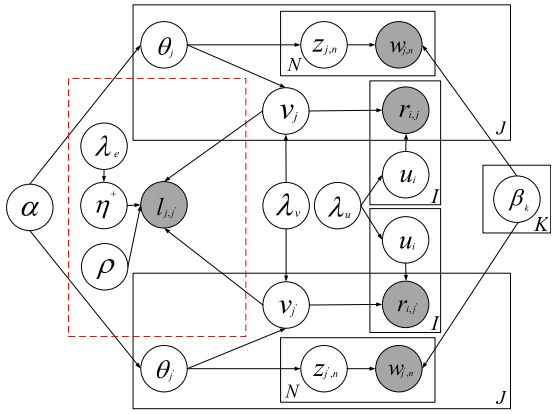


Fig. 3. The graphical model of the degenerated RCTR. The component in the dashed rectangle is what differentiates RCTR from CTR.

d) For each word in the document (item) $\mathbf{w}_j$, which is denoted as w_{jn},
   i) Draw topic assignment $z_{jn} \sim \text{Mult}(\theta_j)$.
   ii) Draw word $w_{jn} \sim \text{Mult}(\beta_{z_{jn}})$.

3) Draw the parameter $\eta^+ \sim \mathcal{N}(0, \lambda_e^{-1} I_{K+1})$.

4) For each pair of items (j, j'), draw a binary link indicator

$$l_{j,j'} | s_j, s_{j'} \sim \psi(\cdot | s_j, s_{j'}, \eta^+).$$

5) Draw the feedback r_{ij} for each user-item pair (i, j),

$$r_{ij} \sim \mathcal{N}(u_i^T v_j, c_{ij}^{-1}).$$

In the above generative process, the *link probability function* is defined as follows:

$$\psi(l_{j,j'} = 1 | s_j, s_{j'}, \eta^+) = \left[\sigma\left(\eta^T (s_j \circ s_{j'}) + \nu\right)\right]^{\rho}, \tag{3}$$

where $l_{j,j'}$ is a binary value, $l_{j,j'} = 1$ means there exists a relation (link) between item j and item j', otherwise $l_{j,j'} = 0$, ν is a scalar value representing the offset, $\eta^+ = \langle \eta, \nu \rangle$ with the symbol $\langle \cdot \rangle$ being the vector-scalar concatenation, the operator $\circ$ means element-wise vector multiplication, and $\sigma(\cdot)$ represents the sigmoid function which is defined as follows:

$$\sigma(x) = \frac{\exp(x)}{1 + \exp(x)}.$$

Steps 2(c), 3, and 4 in the above generative process are what differentiates RCTR from CTR. The *item relational offset* τ_j in Step 2(c) is one of the key properties in RCTR. Similar to the item latent offset, τ_j makes it possible for s_j to diverge from the item latent factor v_j if necessary. While the item latent vector v_j represents what the users think item j is about, the item relational vector s_j represents the effect other items have on item j. The larger λ_r is, the more likely it is that v_j and s_j are close to each other. When λ_r goes to infinity, the model degenerates to the case with $v_j = s_j$, which is shown in Fig. 3. In the experiments which are presented in Section 4 (Fig. 21), we show that the RCTR model in Fig. 2 outperforms the degenerated model in Fig. 3, which verifies the effectiveness of the item relational offset τ_j. Note that for simplicity and fair comparison, we use the same Gaussian feedback model in Step 5 as in [45]. Since the feedback is binary, it is possible to further improve the performance by using a Logistic feedback model.

3.2 Learning

Based on the RCTR model in Fig. 2, we may treat all the parameters as random variables and resort to fully Bayesian methods, such as Markov chain Monte Carlo (MCMC) methods and variational methods [23], for learning and inference. However, we do not adopt fully Bayesian methods here for RCTR because these methods typically incur high computational cost. Furthermore, because the focus of this paper is to illustrate the importance of the relations among items which are not utilized in CTR, it is more reasonable to adopt the same learning strategy as that in CTR for learning and inference.

Hence, as that in CTR, a maximum a posteriori estimate is also adopted for parameter learning in RCTR. The MAP estimate tries to maximize the log-posteriori of U, V, η^+, $s_{1:J}$, $\theta_{1:J}$, and $\boldsymbol{\beta}$ given the hyper-parameters ρ, λ_u, λ_v, λ_r, and λ_e

$$\begin{aligned}
\mathscr{L} =& \rho \sum_{l_{j,j'}=1} \log \sigma(\eta^T (s_j \circ s_{j'}) + \nu) - \frac{\lambda_u}{2} \sum_i u_i^T u_i \\
& - \frac{\lambda_v}{2} \sum_j (v_j - \theta_j)^T (v_j - \theta_j) \\
& - \frac{\lambda_r}{2} \sum_j (s_j - v_j)^T (s_j - v_j) - \frac{\lambda_e}{2} \eta^{+T} \eta^+ \\
& + \sum_j \sum_n \log\left(\sum_k \theta_{jk} \beta_{k, w_{jn}} \right) \\
& - \sum_{i,j} \frac{c_{ij}}{2} \left(r_{ij} - u_i^T v_j\right)^2.
\end{aligned} \tag{4}$$

A constant is omitted and the hyper-parameter of the topic model α is set to 1 as in CTR. This objective function can be optimized using coordinate ascent. Since $\mathscr{L}$ is not jointly convex in all the variables, we design an alternating algorithm to learn the parameters. More specifically, each time we optimize one parameter with all other parameters fixed.

For u_i and v_j, by setting the gradient to zero, we get the following update rules:

$$\begin{aligned}
u_i &\leftarrow (V C_i V^T + \lambda_u I_K)^{-1} V C_i R_i, \\
v_j &\leftarrow (U C_i U^T + \lambda_v I_K + \lambda_r I_K)^{-1} (U C_j R_j + \lambda_v \theta_j + \lambda_r s_j),
\end{aligned}$$

where C_i is a diagonal matrix with $\{c_{ij}|j = 1, , J\}$ being its diagonal entries, and $R_i = \{r_{ij}|j = 1, 2, \ldots, J\}$ is a column vector containing all the feedbacks of user i. Note that c_{ij} is defined in (2), which reflects the confidence controlled by a and b, as discussed in [21].

For s_j and η^+, since we can not directly take the gradients of $\mathscr{L}$ with respect to s_j or η^+ and set them to zero, gradient ascent is used to update the variables. The gradient of $\mathscr{L}$ with respect to s_j is

$$\nabla_{s_j}\mathscr{L} = \rho \sum_{l_{j,j'}=1} (1 - \sigma(\eta^T(s_j \circ s_{j'}) + \nu))(\eta \circ s_{j'}) - \lambda_r(s_j - v_j).$$

The gradient of $\mathscr{L}$ with respect to η^+ is

$$\nabla_{\eta^+}\mathscr{L} = \rho \sum_{l_{j,j'}=1} (1 - \sigma({\eta^+}^T \pi^+_{j,j'}))\pi^+_{j,j'} - \lambda_e \eta^+,$$

where we denote $\pi^+_{j,j'} = \langle s_j \circ s_{j'}, 1\rangle$.

For θ_j, we first define $q(z_{jn} = k) = \psi_{jnk}$ as seen in [45], and then apply Jensen's inequality after the items containing θ_j are separated,

$$\begin{aligned}\mathscr{L}(\theta_j) &\geq -\frac{\lambda_v}{2}(v_j - \theta_j)^T(v_j - \theta_j) \\ &\quad + \sum_n \sum_k \phi_{jnk}(\log \theta_{jk}\beta_{k,w_{jn}} - \log \phi_{jnk}) \\ &= \mathscr{L}(\theta_j, \boldsymbol{\phi}_j).\end{aligned}$$

Here $\boldsymbol{\phi}_j = (\phi_{jnk})^{N\times K}_{n=1,k=1}$. Obviously $\mathscr{L}(\theta_j, \boldsymbol{\phi}_j)$ is a tight lower bound of $\mathscr{L}(\theta_j)$ and we can use projection gradient to optimize θ_j. The optimal ϕ_{jnk} is

$$\phi_{jnk} \propto \theta_{jk}\beta_{k,w_{jn}}.$$

As for the parameter β, we follow the same M-step update as in LDA,

$$\beta_{kw} \propto \sum_j \sum_n \phi_{jnk}\mathbf{1}[w_{jn} = w].$$

3.3 Prediction

Let D be the observed test data. Similar to [45], we use a point estimate of u_i, θ_j and ϵ_j to calculate the predicted feedback

$$E[r_{ij}|D] \approx E[u_i|D]^T(E[\theta_j|D] + E[\epsilon_j|D]),$$

where $E(\cdot)$ denotes the expectation operation.

For in-matrix prediction

$$r^*_{ij} \approx (u^*_j)^T(\theta^*_j + \epsilon^*_j) = (u^*_i)^T v^*_j.$$

For out-of-matrix prediction, since $E[\epsilon_j] = 0$, we have

$$r^*_{ij} \approx (u^*_i)^T\theta^*_j.$$

3.4 Time Complexity

According to the update rules in the RCTR learning procedure, we can see that for each iteration the time complexity for updating η is $O(KL)$ where K is the dimensionality of the latent factor space and L is the total number of relations (links) in the relation network. The cost of updating the item relational matrix $S = \{s_j|j = 1, 2, \ldots, J\}$ is also $O(KL)$ for each iteration. The complexity for updating other variables is the same as that in CTR. For U, the time complexity is $O(IK^3 + IJK^2)$ and for V it is $O(JK^3 + IJK^2)$ where I is the number of users and J is the number of items. In each iteration, RCTR only adds $O(KL)$ of extra time complexity to CTR. Since the relation network is typically sparse which means L can be treated as a constant multiple of J, the extra time cost in RCTR is minimal.

From our experiments, we find that RCTR needs a smaller number of learning iterations than CTR to achieve satisfactory accuracy. As a consequence, the total empirical measured runtime of training RCTR is lower than that of training CTR even if the time complexity of each iteration of RCTR is slightly higher than that of CTR. This is verified in the experimental results in Section 4.7.

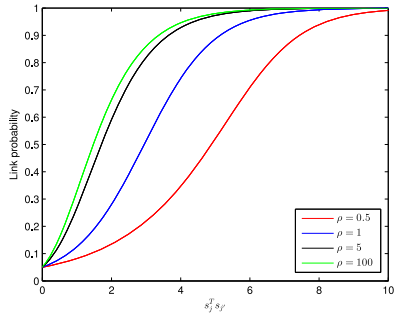


Fig. 4. A comparison of link probability functions with different ρ. The curves plot the probability of $l_{j,j'} = 1$ as a function of the inner product of s_j and $s_{j'}$. η is set to 1 and ν is adapted so that all functions have the same starting point.

3.5 Discussion on Link Probability Function

Another key property of RCTR is the family of link probability functions, which is inspired by the relational topic model (RTM) [14]. The authors in RTM [14] find that different link probability functions can achieve different prediction performance. In RCTR, we use a single parameter ρ to control the choice of the link probability function. Since ρ is a non-negative real number, the family of link probability functions actually contains an infinite number of candidate link probability functions. However, only two link probability functions are proposed in [14]. Hence, our new family of link probability functions can increase the modeling capacity of RCTR, and consequently better performance can be expected. From the perspective of optimization, ρ can simply be regarded as a necessary regularization hyper-parameter to control the tradeoff between relations and other observations, which can easily be seen from (4). Comparison between link probability functions with different ρ is shown in Fig. 4, from which we can see that our link probability functions are flexible enough to model different cases.

Note that if $\rho = 1$, our link probability function degenerates to one of the link probability functions mentioned in RTM [14]. Moreover, when $\rho = 1$, $\nu = 0$ and η is a vector with all elements being one, $\psi(l_{j,j'} = 1|s_j, s_{j'}, \eta^+) = \sigma(s_j^T s_{j'})$, which is the link probability function in CTR-SMF [31]. The

parameters in ν and η make our link probability function more flexible than those in CTR-SMF. In our experiments we find that the elements in the learned vector η are not identical and $\nu \neq 0$, which means that parameters η and ν are necessary for differentiating the importance of different elements of the latent vector in deciding whether or not two items are linked.

4 Experiments

We design several experiments and compare the prediction performance between RCTR and the state-of-the-art methods on two real-world datasets. The questions we are trying to answer are:

- To what degree does RCTR outperform the state-of-the-art methods, especially when the data is extremely sparse?
- To what degree does the family of link probability functions help improve the prediction performance?
- How is the prediction performance affected by the relational parameter λ_r and other parameters?

4.1 Datasets

We use two real-world datasets to conduct our experiments. Both of them are from CiteULike,[2] but they are collected in different ways with different scales and degrees of sparsity. For the feedback matrix in the datasets, if a user reads (or posts) a paper, the corresponding feedback is 1. Otherwise, if a user has not read (or posted) a paper, the corresponding feedback is missing (denoted by 0). The first dataset, *citeulike-a*, is from [45]. Note that the original dataset in [45] does not contain relations between items. We collect the items' relational information from CiteULike and Google Scholar.[3] The second dataset, *citeulike-t*, we collect independently from the first one. We manually select 273 seed tags and collect all the articles with at least one of these tags. We also crawl the citations between the articles from Google Scholar. Note that the final number of tags associated with all the collected articles is far more than the number (273) of seed tags. Similar to [45], we remove any users with fewer than three articles. The description of these two datasets is shown in Table 1. We can see that the number of users and items in our collected *citeulike-t* dataset is larger than that of *citeulike-a*. Furthermore, the ratios of non-missing entries (equal to 1—*sparsity*) in the user-item matrices of *citeulike-a* and *citeulike-t* are 0.0022 and 0.0007 respectively, which means that the second dataset is sparser than the first.

The text information (item content) of *citeulike-a* is preprocessed by following the same procedure as that in [45] and we also use their articles' titles and abstracts for the text information of *citeulike-t*. After removing the stop words, we choose the top 20,000 discriminative words according to the tf-idf values as our vocabulary.

TABLE 1
Description of Datasets

	citeulike-a	citeulike-t
#users	5,551	7,947
#items	16,980	25,975
#tags	19,107	52,946
#citations	44,709	32,565
#user-item pairs	204,987	134,860
sparsity	99.78%	99.93%
#relations	549,447	438,722

Besides the citation relations between papers, we find that the tags associated with papers are also an informative signal about the relations between papers. Hence, we use both citations and tags to construct a single relation network (graph) for each dataset. For each dataset, we first construct a *tag graph* with a threshold of 4, that is, if two papers have four or more tags in common, they are linked in the tag graph. Then we impose an OR operation on the *tag graph* and *citation graph* to get the final *relation graph* for the dataset. More specifically, if a relation exists in either the *tag graph* or the *citation graph*, it also exists in the final *relation graph*. The final numbers of relations (links) in the relation graphs are shown in the last row of Table 1.

4.2 Evaluation Scheme

We design evaluation schemes to evaluate models in both user-oriented and item-oriented settings.

For the user-oriented setting:

- Select some percentage Q (e.g. 10 percent) of the users as test users. The training set contains two parts: one part includes all feedbacks of the other (1—Q) of the users, and the other part includes P positive feedbacks (with value 1) for each test user.
- Perform prediction on the remaining feedbacks of the test users.
- Repeat the above procedure for $1/Q$ rounds. For each round, we select different test users. For example, if $Q = 10\%$, we perform $1/Q = 10$ rounds of tests. This is equivalent to a 10-fold cross validation procedure where each user appears one time in a test set. If P is small, the test set actually contains some new users with little feedback.

We evaluate the predictive performance with two cases: $Q = 10\%$ and $Q = 100\%$. The case $Q = 10\%$ means that the recommendation system has been running for a long time and only a small number of users are new. The case $Q = 100\%$ means that the system is online for only a while and most of the users are new. As stated in Section 1, providing a good recommendation for new users with little feedback is more important than that for frequent users. Hence, it is more interesting to study the performance of recommendation algorithms in extremely sparse settings. We let P vary from 1 to 10 in our experiments and the smaller the P, the sparser the training set. Note that when $P = 1$ and $Q = 100\%$, only 2.7 percent of the entries with

2. CiteULike allows users to create their own collections of articles. There are abstracts, titles, and tags for each article. Other information like authors, groups, posting time, and keywords is not used in this paper. The details can be found at http://www.citeulike.ort/faq/data.adp.

3. Most of the articles in CiteULike do not provide citation information, which means we have to collect them by ourselves. In this paper, the citation information is collected from Google Scholar: http://scholar.google.com.

value 1 are put in the training set for dataset *citeulike-a* and the number for dataset *citeulike-t* is 5.8 percent.

As in [45] and [31], we use recall as our evaluation metric since zero feedback may be caused either by users who dislike an item or by users who do not know the existence of the item, which means precision is not a proper metric here. Like most recommender systems, we sort the predicted feedback of candidate items which are any remaining items that are not put into the training data, and recommend the top M items (articles) to the target user. For each user, recall@M is defined as

$$\text{recall@M} = \frac{\text{number of items the user likes in top M}}{\text{total number of items the user likes}}.$$

The final reported result is the average of all the users' recall.

The above evaluation scheme is for the user-oriented setting. We can also use a similar evaluation scheme for the item-oriented setting where we need to recommend users to target items. Under this setting, the item-oriented recall i-recall@M can be defined as follows:

$$\text{i} - \text{recall@M} = \frac{\text{number of users interested in the item in top M}}{\text{total number of users interested in the item}}.$$

4.3 Baselines and Experimental Settings

The models we used for comparison are listed as follows:

- *MP*. The most-popular baseline which orders users or items by how often they appear in the training set.
- *MC*. The most-cited baseline which orders items (papers) by how often they are cited in the user-oriented setting. For the item-oriented setting, the MC baseline will order the users by the total number of citations of the items (papers) rated by each user.
- *CF*. The CF based on MF [25]. We use the same variant of [25] as that in [45]. It is essentially a probabilistic formulation of [25].
- *CTR*. The CTR model [45] with the text information as the input of the LDA.
- *RCTR*. The RCTR model with the text information as the input of the LDA.
- *CT+*. A variant of content-based methods to incorporate the citation and tag information. We first construct a dictionary containing the original words from the text information and the citations and tags as additional words. The bag-of-words of an article is used as its feature vector. The feature vector of a user is calculated as the average of the feature vectors of the articles s/he gave feedback to. We recommend the items to users with the largest cosine similarities.
- *CTR+*. The CTR model with the citations and tags as additional words for the input of the LDA.
- *RCTR+*. The RCTR model with the citations and tags as additional words for the input of the LDA.

In the experiments, we first use a validation set to find the optimal parameters λ_u and λ_v for CTR and CF. Then we fix λ_u and λ_v to the optimal values for CTR and vary the other parameters. By using five-fold cross validation on the training set with grid search, we find that both CF and CTR achieve good performance when $\lambda_v = 100$, $\lambda_u = 0.01$, $a = 1$, $b = 0.01$. As in CTR [45], we choose K to be 200. Here a and b control the confidence parameters c_{ij}. For RCTR, we set the parameters $\lambda_v = 100$, $\lambda_u = 0.01$, $a = 1$, $b = 0.01$, $K = 200$ and vary the other parameters, including the parameter ρ that controls the link probability function, to see how the choice of *link probability function* affects the prediction performance and study the sensitivity to parameters λ_r and λ_e.

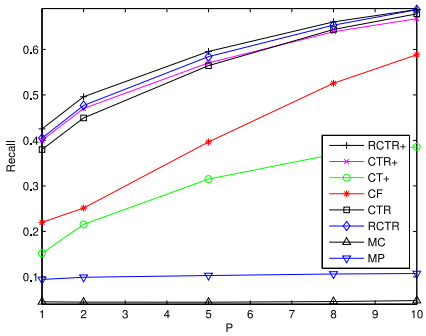


Fig. 5. User-oriented recall@300 when $Q = 10\%$ and P ranges from 1 to 10 on dataset *citeulike-a*.

As in CTR, we choose M to be 50, 100, 150, 200, 250, 300 in recall@M and i-recall@M. Although a smaller M might be more reasonable in some applications, there also exist other scenarios where a large M makes sense. For example, more than 100 papers should typically be read by a PhD student when he does a literature survey about his thesis topic. Hence, M is set to be no smaller than 50 in this paper.

4.4 Performance

As stated in Section 4.2, we have two different recommendation settings: user-oriented recommendation and item-oriented recommendation. We evaluate both in this section.

4.4.1 User-Oriented Recommendation

User-oriented recommendation tries to recommend items to target users. Fig. 5 shows the recall@300 on dataset *citeulike-a* when $Q = 10\%$ and P is set to be 1, 2, 5, 8, 10.[4] We can see that the baselines MP and MC perform poorly. For all other settings, MP and MC always achieve the worst performance. To avoid clutter and better demonstrate the differences between other stronger models, we choose to drop the corresponding lines for baselines MC and MP in the following experiments. Fig. 6 shows the recall@300 on dataset *citeulike-t* when $Q = 10\%$ by changing P. Figs. 7 and 8 show the recall@300 when $Q = 100\%$ with different P, on *citeulike-a* and *citeulike-t*, respectively. From these figures, we can find that CF performs poorly when P is extremely small and gets closer to CTR when P gets larger. The performance of CF is far from satisfactory due to the sparsity problem. CTR can outperform CF by using item content

4. Note that the standard deviations for all the settings are very small (lie in the range $[5.73 \times 10^{-5}, 8.35 \times 10^{-3}]$). For better illustration of the figures, we do not separately report the standard deviations for all the figures in this paper.

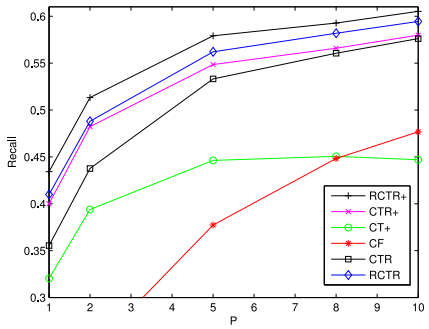


Fig. 6. User-oriented recall@300 when $Q = 10\%$ and P ranges from 1 to 10 on dataset *citeulike-t*.

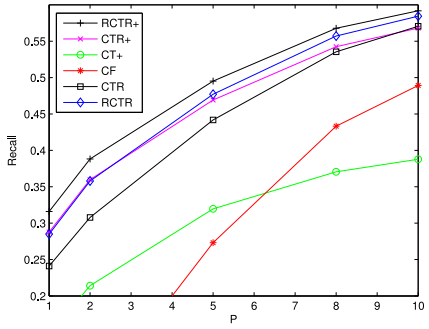


Fig. 7. User-oriented recall@300 when $Q = 100\%$ and P ranges from 1 to 10 on dataset *citeulike-a*.

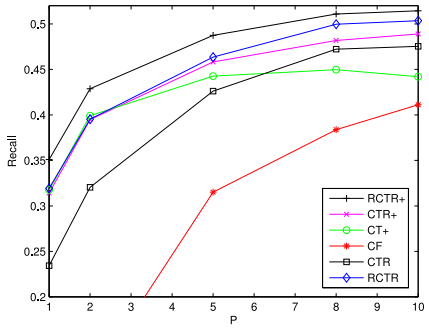


Fig. 8. User-oriented recall@300 when $Q = 100\%$ and P ranges from 1 to 10 on dataset *citeulike-t*.

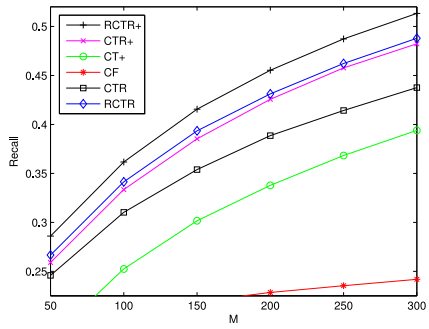


Fig. 9. User-oriented recall@M when M ranges from 50 to 300 on dataset *citeulike-t*. $Q = 10\%$ and P is set to 2.

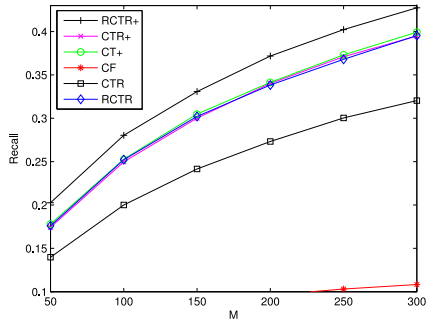


Fig. 10. User-oriented recall@M when M ranges from 50 to 300 on dataset *citeulike-t*. $Q = 100\%$ and P is set to 2.

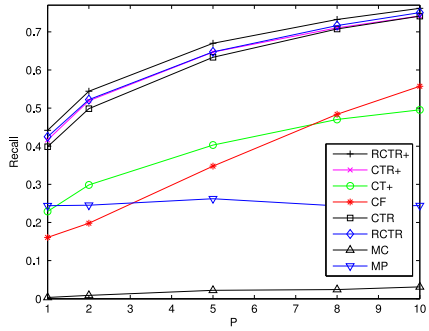


Fig. 11. Item-oriented i-recall@300 when $Q = 10\%$ and P ranges from 1 to 10 on dataset *citeulike-a*.

information, and our RCTR can further improve the performance by effectively integrating the item relations into modeling. In most cases, the CTR and RCTR can outperform the content-based method CT+. Adding the tags and relations as extra words means CTR+ can achieve a better performance than CTR, and in some cases CTR+ being comparable to RCTR. Note that although both CTR+ and RCTR use the same information for modeling, part of the tag information will be lost in RCTR because RCTR uses tags for graph construction and only a tag-relation is kept if two items have four or more tags in common. Hence, it is more fair to compare CTR+ with RCTR+. We find our RCTR+ outperforms CTR+ in all cases although RCTR+ uses the same information as CTR+, which verifies the effectiveness of our modeling procedure.

Figs. 9 and 10 show the recall of the models on dataset *citeulike-t* when we set P to 2 and M = 50, 100, 150, 200, 250, 300. A similar phenomenon can be observed for dataset *citeulike-a* and other values of P, which are omitted to save space. For RCTR, $\lambda_r = 1$, $\lambda_e = 1{,}000$, and $\rho = 100$. Again RCTR+ achieves the best performance, and RCTR is consistently better than CTR and CF.

4.4.2 Item-Oriented Recommendation

Item-oriented recommendation tries to recommend users to target items. In the scientific article area, it can be used for applications like recommending coauthors or reviewers. Fig. 11 shows the i-recall@300 on dataset *citeulike-a* when $Q = 10\%$ and P is set to be 1, 2, 5, 8, 10. Similar to the user-oriented case, we can see that baselines MP and MC perform poorly. For all other settings, MP and MC always achieve the worst performance. To avoid clutter and better demonstrate the differences between other stronger models, we choose to drop the corresponding lines for baselines MC and MP in the following experiments. Fig. 12 shows the i-recall@300 on dataset *citeulike-t* when $Q = 10\%$ with different values of

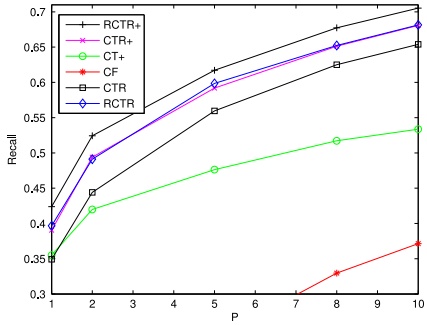


Fig. 12. Item-oriented i-recall@300 when $Q = 10\%$ and P ranges from 1 to 10 on dataset *citeulike-t*.

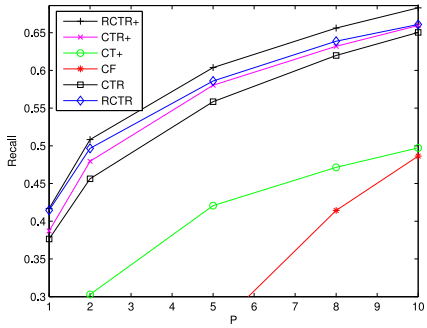


Fig. 13. Item-oriented i-recall@300 when $Q = 100\%$ and P ranges from 1 to 10 on dataset *citeulike-a*.

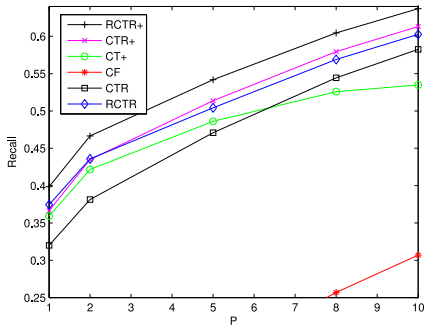


Fig. 14. Item-oriented i-recall@300 when $Q = 100\%$ and P ranges from 1 to 10 on dataset *citeulike-t*.

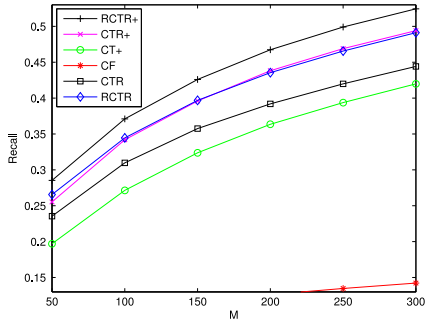


Fig. 15. Item-oriented i-recall@M when M ranges from 50 to 300 on dataset *citeulike-t*. $Q = 10\%$ and P is set to 2.

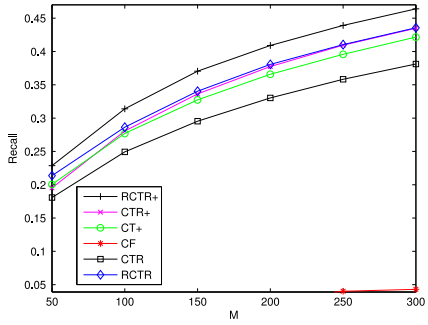


Fig. 16. Item-oriented i-recall@M when M ranges from 50 to 300 on dataset *citeulike-t*. $Q = 100\%$ and P is set to 2.

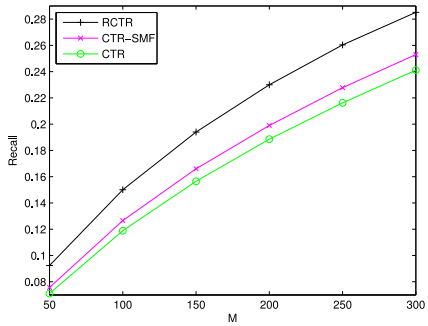


Fig. 17. User-oriented recall@M when M ranges from 50 to 300 on dataset *citeulike-a*. $Q = 100\%$ and P is set to 1.

P. Figs. 13 and 14 show the i-recall@300 on *citeulike-a* and *citeulike-t* when $Q = 100\%$, respectively.

Figs. 15 and 16 show the item-oriented recall on dataset *citeulike-t* when we fix P to be 2 and set M = 50, 100, 150, 200, 250, 300.

From the figures, we find that in the item-oriented setting RCTR can also outperform CTR and CF in most cases, and once again RCTR+ achieves the best performance.

4.5 Comparison to the Adapted CTR-SMF

As mentioned in previous sections, CTR-SMF extends CTR by using social networks among *users* to improve performance. Hence, CTR-SMF cannot be directly used for any of application scenarios in this paper. Here, we adapt the original CTR-SMF in [31] for our scenarios by adopting the same techniques in CTR-SMF to model the item network. More specifically, compared with RCTR, the adapted CTR-SMF adopts a simpler link function as stated in Section 3.5. Furthermore, the adapted CTR-SMF does not adopt relational offsets which are key parts of RCTR. Figs. 17 and 18 show the results of CTR, CTR-SMF and RCTR. We find that the adapted CTR-SMF outperforms CTR, and our RCTR outperforms CTR-SMF.

4.6 Sensitivity to Parameters

Fig. 19 shows how the choice of link probability functions (depending on ρ) affect the prediction performance. We set $\lambda_r = 1$, $\lambda_e = 1{,}000$, and $P = 1$, and then calculate the recall@M by setting $\rho = 1, 10, 100, 1{,}000, 10{,}000$. When $\rho = 1$, the link probability function is identical to the one proposed in RTM [14]. We can see that $\rho = 100$ is the optimal link probability function among the five. When ρ is too small, the performance is close to that of CTR as expected. Note that the link probability function proposed in RTM [14] has the worst prediction performance, which demonstrates the importance of choosing the optimal link

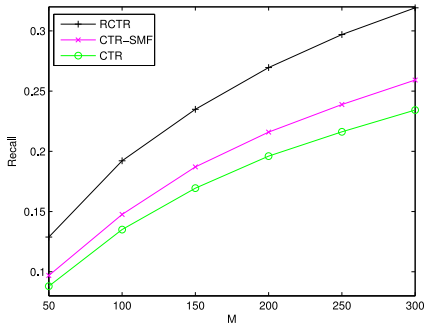


Fig. 18. User-oriented recall@M when M ranges from 50 to 300 on dataset *citeulike-t*. $Q = 100\%$ and P is set to 1.

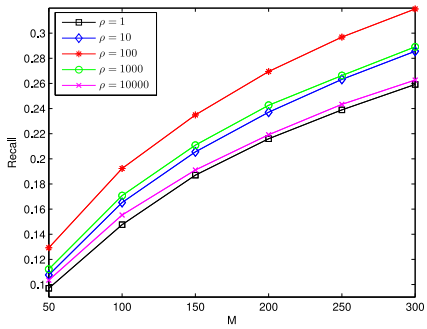


Fig. 19. The effect of ρ in RCTR when M ranges from 50 to 300 on dataset *citeulike-t*. $Q = 100\%$ and P is set to 1. $\lambda_v = 100$, $\lambda_u = 0.01$, $\lambda_r = 1$, $\lambda_e = 1{,}000$.

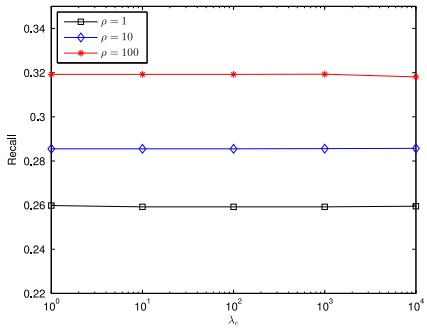


Fig. 20. The effect of λ_e in RCTR on dataset *citeulike-t*. $Q = 100\%$ and P is set to 1. $\lambda_v = 100$, $\lambda_u = 0.01$, and $\lambda_r = 1$.

TABLE 2
Number of Iterations for Training CTR and RCTR on Dataset *citeulike-t*

P	1	2	5	8	10
CTR	48.8 ± 2.9	45.8 ± 2.7	73.4 ± 1.0	86.0 ± 1.8	87.0 ± 2.1
RCTR	14.6 ± 1.6	14.2 ± 1.4	19.6 ± 1.0	18.8 ± 1.1	19.6 ± 1.0

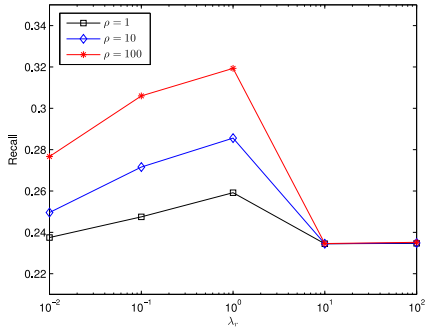


Fig. 21. The effect of λ_r in RCTR on dataset *citeulike-t*. $Q = 100\%$ and P is set to 1. $\lambda_v = 100$, $\lambda_u = 0.01$, and $\lambda_e = 1{,}000$.

probability function. Our RCTR provides a flexible family of functions for choice.

To examine the sensitivity of RCTR to parameters λ_r and λ_e, we conduct two sets of experiments with the training data of $P = 1$. First, we set $\lambda_r = 1$ and see how the performance changes with λ_e. Fig. 20 shows the recall@300 on the dataset *citeulike-t*. We find that RCTR is stable with the change of λ_e. We then set $\lambda_e = 1{,}000$ and see how λ_r affects the prediction performance. Fig. 21 shows the recall@300 on the dataset *citeulike-t*. We can see that the performance is relatively sensitive to λ_r. For a fixed ρ, the recall first increases with λ_r and then decreases after somewhere near $\lambda_r = 1$. The performance remains relatively poor (near 24 percent) when λ_r is too large for all three choices of the link probability functions in the experiment. A larger λ_r means item relational vectors s_j will be closer to item latent vectors v_j. When $\lambda_r = 0$ RCTR degenerates to CTR and when $\lambda_r = \infty$, RCTR degenerates to the model in Fig. 3. The nice property is that the best performance is always achieved near $\lambda_r = 1$ when we change other variables like ρ. Hence, we can fix $\lambda_r = 1$ in our experiment.

4.7 Computational Cost

Table 2 shows the number of iterations for training CTR and RCTR on dataset *citeulike-t*. Table 3 shows the average empirical measured time (in seconds) for training CTR and RCTR on dataset *citeulike-t*. As stated in Section 3.4, although more information is used and then more time is needed for each iteration of training in RCTR, the total empirical measured time for training RCTR is still lower than that for training CTR. The main reason is that RCTR needs a smaller number of learning iterations than CTR to achieve satisfactory accuracy.

Note that the time in Table 3 is measured when $K = 200$ which is the same setting as CTR in [45]. In our experiments, we find that the accuracy of $K = 50$ is comparable to that of $K = 200$. As mentioned in Section 3.4, the time complexity for one iteration is cubic with respect to K. Hence, if we choose K to be 50, the empirical measured time would be about $1/64$ of the time in Table 3, which is small.

TABLE 3
Empirical Measured Time for Training CTR and RCTR on Dataset *citeulike-t* (in Seconds)

P	1	2	5	8	10
CTR	$16{,}250 \pm 1{,}114$	$15{,}251 \pm 1{,}010$	$24{,}442 \pm 379$	$28{,}638 \pm 706$	$28{,}971 \pm 706$
RCTR	$5{,}285 \pm 657$	$5{,}140 \pm 594$	$7{,}095 \pm 412$	$6{,}806 \pm 472$	$7{,}095 \pm 412$

TABLE 4
Interpretability of the Learned Latent Structures

	user I (RCTR)	in user's lib?
top 3 topics	1. activity, neural, neurons, cortex, cortical, neuronal, stimuli, spike, visual, stimulus 2. processing, conditions, sensitivity, perception, music, sound, filters, filter, simultaneous, auditory 3. positive, correlation, hypothesis, negative, correlations, bias, intrinsic, costs, codon, aggregation	
top 10 articles	1. The variable discharge of cortical neurons	yes
	2. Refractoriness and neural precision	no
	3. Neural correlates of decision variables in parietal cortex	yes
	4. Neuronal oscillations in cortical networks	no
	5. Synergy, redundancy, and independence in population codes	yes
	6. Entropy and information in neural spike trains	yes
	7. The Bayesian brain: the role of uncertainty in neural coding and computation	yes
	8. Activity in posterior parietal cortex is correlated with the relative subjective desirability of action	yes
	9. Psychology and neurobiology of simple decisions	yes
	10. Role of experience and oscillations in transforming a rate code into a temporal code	yes
	user I (CTR)	in user's lib?
top 3 topics	1. coding, take, necessary, place, see, regarding, reason, recognized, mediated, places 2. genetic, variation, population, populations, variants, snps, individuals, genetics, phenotypes, phenotypic 3. activity, neural, neurons, cortex, cortical, neuronal, stimuli, spike, visual, stimulus	
top 10 articles	1. Chromatin modifications and their function	no
	2. Mistranslation-induced protein misfolding as a dominant constraint on coding-sequence evolution	no
	3. Lateral habenula as a source of negative reward signals in dopamine neurons	yes
	4. Two types of dopamine neuron distinctly convey positive and negative motivational signals	no
	5. Proportionally more deleterious genetic variation in European than in African populations	no
	6. The primate amygdala represents the positive and negative value of visual stimuli during learning	yes
	7. Genetic variation in an individual human exome	no
	8. Behavioural report of single neuron stimulation in somatosensory cortex	no
	9. Reward-dependent modulation of neuronal activity in the primate dorsal raphe nucleus	no
	10. Uniform inhibition of dopamine neurons in the ventral tegmental area by aversive stimuli	yes
	user II (CTR)	in user's lib?
top 3 topics	1. sites, target, site, targets, mirnas, predicted, mirna, conserved, seed, figure 2. rna, mirnas, mirna, rnas, mrna, micrornas, microrna, translation, mir, mrnas 3. human, identification, humans, targeted, curves, curve, assay, roc, uniquely, receiver	
top 10 articles	1. Combinatorial microRNA target predictions	yes
	2. Prediction of mammalian microRNA target	yes
	3. Conserved seed pairing indicates that thousands of human genes are microRNA targets	yes
	4. Animal microRNAs confer robustness to gene expression	yes
	5. Silencing of microRNAs in vivo with 'antagomirs'	no
	6. Microarray analysis shows that some microRNAs downregulate large numbers of target mRNAs	yes
	7. Mammalian microRNAs derived from genomic repeats	yes
	8. Identification of hundreds of conserved and nonconserved human microRNAs	yes
	9. The widespread impact of mammalian microRNAs on mRNA repression and evolution	yes
	10. A microRNA polycistron as a potential human oncogene	yes
	user II (CTR)	in user's lib?
top 3 topics	1. rna, mirnas, mirna, rnas, mrna, micrornas, microrna, translation, mir, mrnas 2. sites, target, site, targets, mirnas, predicted, mirna, conserved, seed, figure 3. human, identification, humans, targeted, curves, curve, assay, roc, uniquely, receiver	
top 10 articles	1. Regulation by let-7 and lin-4 miRNAs results in target mRNA degradation	no
	2. Getting to the root of miRNA-mediated gene silencing	no
	3. Prediction of mammalian microRNA target	yes
	4. Conserved seed pairing indicates that thousands of human genes are microRNA targets	yes
	5. Animal microRNAs confer robustness to gene expression	yes
	6. Large-scale sequencing reveals 21U-RNAs and additional microRNAs and endogenous siRNAs in C. elegans	no
	7. MicroRNA control in the immune System: basic principles	no
	8. Concordant regulation of translation and mRNA abundance for hundreds of targets of a human microRNA	no
	9. Endogenous siRNAs from naturally formed dsRNAs regulate transcripts in mouse oocytes	yes
	10. Dual role for argonautes in microRNA processing	no

4.8 Interpretability

The learned results of RCTR have a good interpretation. More specifically, the latent vectors of the users can be interpreted as topics learned from the data. To gain a better insight into RCTR, we present two example users' profiles which correspond to their top three matched topics and the recommended articles returned by RCTR and CTR. In this case study, we train the RCTR and CTR using the sparse training data with $P = 1$ before recommending articles for the users. Note that in the training data, each user has only given feedback to one single article, which makes recommendation challenging. As shown in Table 4, user I is a researcher working on neural science, which is clearly indicated in the first topic returned by RCTR and the third one returned by CTR. The precision of the top 10 articles for RCTR and CTR are 80 and 30 percent respectively for user I. Similarly, we find that user II is a biologist with a search focusing on RNA. The precision of RCTR and CTR are 90 and 40 percent, respectively.

If we examine the training set, we find that user I gave feedback to only one article with the title *'Neural Correlations, Population Coding and Computation'*. Among the eight correctly recommended articles returned by RCTR, six of them are directly connected (by relation) to the rated article *'Neural Correlations, Population Coding and Computation'* in the relation graph, which indicates that RCTR successfully incorporates relation information into the model and achieves a boost in prediction performance. Similarly, the article given feedback by user II in the training set is *'A Combined Computational-Experimental Approach Predicts Human MicroRNA Targets'*. Among the nine correctly returned articles, four of them are connected (by relation) to the one rated in the relation graph. More specifically, three of them link to *'A Combined Computational-Experimental Approach Predicts Human MicroRNA Targets'* in both the *tag graph* and the *citation graph* and one of them links to it only in the *tag graph*.

5 Conclusion and Future Work

In this paper, we have developed a novel hierarchical Bayesian model called RCTR for recommender systems. RCTR can seamlessly integrate the user-item feedback information, item content information and network structure among items into the same principled model. RCTR can utilize extra information to alleviate the sparsity problem faced by traditional CF methods and CTR. Experiments on real-world datasets show that our model achieves better prediction accuracy than the state-of-the-art methods with a lower empirical training time. Moreover, RCTR has the ability to provide interpretable results which are useful for recommendation.

The Bayesian formulation of RCTR is flexible enough for us to model more than one item network. For example, we can separately model the tag graph and citation graph instead of combining them into one single graph by introducing other latent variables. We can also adapt our RCTR model for the CTR-SMF setting with social networks among users. Moreover, it is easy to design some distributed learning algorithms for RCTR, which would make RCTR scalable for big data modeling. The above possible extensions will be pursued in our future work.

Acknowledgments

This work was supported by the NSFC (No. 61100125, No. 61472182), and the 863 Program of China (No. 2012AA011003). Wu-Jun Li is the corresponding author of this paper.

References

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.

[2] D. Agarwal and B.-C. Chen, "Regression-based latent factor models," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 19–28.

[3] D. Agarwal and B.-C. Chen, "fLDA: Matrix factorization through latent dirichlet allocation," in *Proc. 3rd Int. Conf. Web Search Data Mining*, 2010, pp. 91–100.

[4] D. Almazro, G. Shahatah, L. Albdulkarim, M. Kherees, R. Martinez, and W. Nzoukou, "A survey paper on recommender systems," *CoRR*, abs/1006.5278, 2010.

[5] M. Balabanović and Y. Shoham, "Fab: Content-based, collaborative recommendation," *Commun. ACM*, vol. 40, no. 3, pp. 66–72, 1997.

[6] A. B. Barragáns-Martínez, E. Costa-Montenegro, J. C. Burguillo-Rial, M. Rey-López, F. A. Mikic-Fonte, and A. Peleteiro-Ramallo, "A hybrid content-based and item-based collaborative filtering approach to recommend tv programs enhanced with singular value decomposition," *Inf. Sci.*, vol. 180, no. 22, pp. 4290–4311, 2010.

[7] I. Bartolini, Z. Zhang, and D. Papadias, "Collaborative filtering with personalized skylines," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 2, pp. 190–203, Feb. 2011.

[8] J. Bennett and S. Lanning, "The Netflix prize," in *Proc. KDD Cup Workshop*, 2007, pp. 3–6.

[9] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, 2003.

[10] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowl. Based Syst.*, vol. 46, pp. 109–132, 2013.

[11] J. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proc. 4th Conf. Uncertainity Artif. Intell.*, 1998, pp. 43–52.

[12] R. D. Burke, "Hybrid recommender systems: Survey and experiments," *User Model. User Adapted Interaction*, vol. 12, no. 4, pp. 331–370, 2002.

[13] Y. Cai, H. fung Leung, Q. Li, H. Min, J. Tang, and J. Li, "Typicality-based collaborative filtering recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 3, pp. 766–779, Mar. 2014.

[14] J. Chang and D. M. Blei, "Relational topic models for document networks," in *Proc. Int. Conf. Artif. Intell. Stat.*, 2009, pp. 81–88.

[15] K. Choi, D. Yoo, G. Kim, and Y. Suh, "A hybrid online-product recommendation system: Combining implicit rating-based collaborative filtering and sequential pattern analysis," *Electron. Commerce Res. Appl.*, vol. 11, no. 4, pp. 309–317, 2012.

[16] L. M. de Campos, J. M. Fernández-Luna, J. F. Huete, and M. A. Rueda-Morales, "Combining content-based and collaborative recommendations: A hybrid approach based on bayesian networks," *Int. J. Approx. Reasoning*, vol. 51, no. 7, pp. 785–799, 2010.

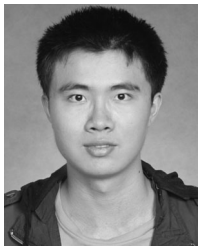
[17] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thieme, "Learning attribute-to-feature mappings for cold-start recommendations," in *Proc. IEEE Int. Conf. Data Mining*, 2010, pp. 176–185.

[18] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 1999, pp. 230–237.

[19] T. Hofmann, "Latent semantic models for collaborative filtering," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 89–115, 2004.

[20] M. F. Hornick and P. Tamayo, "Extending recommender systems for disjoint user/item sets: The conference recommendation problem," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 8, pp. 1478–1490, Aug. 2012.

[21] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proc. IEEE Int. Conf. Data Mining*, 2008, pp. 263–272.
[22] M. Jamali and M. Ester, "TrustWalker: A random walk model for combining trust-based and item-based recommendation," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 397–406.
[23] M. I. Jordan, Z. Ghahramani, T. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," *Mach. Learn.*, vol. 37, no. 2, pp. 183–233, 1999.
[24] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2008, pp. 426–434.
[25] Y. Koren, R. M. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Comput.*, vol. 42, no. 8, pp. 30–37, Aug. 2009.
[26] K. Lang, "Newsweeder: Learning to filter netnews," in *Proc. Int. Conf. Mach. Learn.*, 1995, pp. 331–339.
[27] W.-J. Li and D.-Y. Yeung, "Social relations model for collaborative filtering," in *Proc. AAAI Conf. Artif. Intell.*, 2011, p. 1.
[28] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Comput.*, vol. 7, no. 1, pp. 76–80, Jan. 2003.
[29] H. Ma, H. Yang, M. R. Lyu, and I. King, "SoRec: Social recommendation using probabilistic matrix factorization," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2008, pp. 931–940.
[30] Y.-J. Park, "The adaptive clustering method for the long tail problem of recommender systems," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 8, pp. 1904–1915, Aug. 2013.
[31] S. Purushotham, Y. Liu, and C.-C. J. Kuo, "Collaborative topic regression with social matrix factorization for recommendation systems," in *Proc. Int. Conf. Mach. Learn.*, 2012, pp. 759–766.
[32] S. Rendle, "Factorization machines with libFM," *ACM Trans. Intell. Syst. Technol.*, vol. 3, no. 3, p. 57, 2012.
[33] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proc. Conf. Uncertainity Artif. Intell.*, 2009, pp. 452–461.
[34] A. Said, "Identifying and utilizing contextual data in hybrid recommender systems," in *Proc. ACM Conf. Recommender Syst.*, 2010, pp. 365–368.
[35] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 1257–1264.
[36] R. Salakhutdinov and A. Mnih, "Bayesian probabilistic matrix factorization using markov chain monte carlo," in *Proc. Int. Conf. Mach. Learn.*, 2008, pp. 880–887.
[37] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, "Item-based collaborative filtering recommendation algorithms," in *Proc. Int. World Wide Web Conf.*, 2001, pp. 285–295.
[38] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2002, pp. 253–260.
[39] Y. Shi, M. Larson, and A. Hanjalic, "List-wise learning to rank with matrix factorization for collaborative filtering," in *Proc. ACM Conf. Recommender Syst.*, 2010, pp. 269–272.
[40] S. K. Shinde and U. V. Kulkarni, "Hybrid personalized recommender system using centering-bunching based clustering algorithm," *Expert Syst. Appl.*, vol. 39, no. 1, pp. 1381–1387, 2012.
[41] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Adv. Artif. Intell.*, vol. 2009, p. 4, 2009.
[42] J. Tang, G.-J. Qi, L. Zhang, and C. Xu, "Cross-space affinity learning with its application to movie recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 7, pp. 1510–1519, Jul. 2013.
[43] M. G. Valdez and B. Parra, "A hybrid recommender system architecture for learning objects," in *Proc. Evol. Des. Intell. Syst. Model., Simul. Control*, 2009, pp. 205–211.
[44] D. Q. Vu, A. U. Asuncion, D. R. Hunter, and P. Smyth, "Dynamic egocentric models for citation networks," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 857–864.
[45] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2011, pp. 448–456.
[46] H. Wang and W.-J. Li, "Online egocentric models for citation networks," in *Proc. Int. Joint Conf. Artif. Intell.*, 2013, pp. 2726–2732.
[47] Y. Z. Wei, L. Moreau, and N. R. Jennings, "Learning users' interests by quality classification in market-based recommender systems," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 12, pp. 1678–1688, Dec. 2005.
[48] G.-E. Yap, A.-H. Tan, and H. Pang, "Discovering and exploiting causal dependencies for robust mobile context-aware recommenders," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 7, pp. 977–992, Jul. 2007.
[49] K. Yu, J. D. Lafferty, S. Zhu, and Y. Gong, "Large-scale collaborative prediction using a nonparametric random effects model," in *Proc. Int. Conf. Mach. Learn.*, 2009, pp. 1185–1192.
[50] K. Yu, S. Zhu, J. D. Lafferty, and Y. Gong, "Fast nonparametric matrix factorization for large-scale collaborative filtering," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2009, pp. 211–218.
[51] Y. Zhen, W.-J. Li, and D.-Y. Yeung, "TagiCoFi: Tag informed collaborative filtering," in *Proc. ACM Conf. Recommender Syst.*, 2009, pp. 69–76.



Hao Wang received the BSc degree in computer science from Shanghai Jiao Tong University, China. He is currently working toward the PhD degree in the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. His research interests are in statistical machine learning and data mining.

Wu-Jun Li received the BSc and MEng degrees in computer science from the Nanjing University of China, and the PhD degree in computer science from the Hong Kong University of Science and Technology. He started his academic career as an assistant professor in the Department of Computer Science and Engineering, Shanghai Jiao Tong University. He then joined Nanjing University where he is currently an associate professor in the Department of Computer Science and Technology. His research interests are in statistical machine learning, pattern recognition, and data mining. He is a member of the IEEE.